A Dead-Reckoning Algorithm for Virtual Human Figures*

Tolga K. Capin[1], Igor Sunday Pandzic[2],
Nadia Magnenat Thalmann[2], Daniel Thalmann[1]

[1]Computer Graphics Laboratory (LIG), Swiss Federal Institute of Technology (EPFL)
CH-1015 Lausanne, Switzerland

[2]MIRALAB-CUI, University of Geneva
CH1211 Geneva 4, Switzerland

In networked virtual environments; when the participants are represented by virtual human figures, the articulated structure of the human body introduces a new complexity in the usage of the network resources; because the size of a message needed to represent the body posture is much larger than the one needed for simple, non-articulated objects. This might create a significant overhead in communication, especially as the number of participants in the simulation increases. In addition, the animation should be realistic, as it is easy to recognize anomalies in the virtual human animation. This requires real-time algorithms to decrease the network overhead, while considering body characteristics. The dead-reckoning technique is a way to decrease the number of messages communicated among the participants, and has been used for simple non-articulated objects in popular systems. In this paper, we present the dead-reckoning approach for articulated virtual human figures, and discuss main issues and experimental results.
***Keywords**: distributed virtual environments, virtual human figures, dead-reckoning, Kalman filtering.*

1 Introduction

In multiuser networked virtual environments, it is necessary to provide a natural means of interaction among participants for better collaboration. The participant representation in a networked VE system has several functions: inform the participants' presence to others, identify and differentiate different participants, visualize the participants' position and orientation, direction of interest, and enable communication among participants. Therefore, in most applications, it is important to represent participants by virtual human figures. The participants visualize the environment through the eyes of their virtual actor, and move their virtual body by different means of body control. In addition, introducing the human-like autonomous actors for various tasks increases the level of interaction within the virtual environment. [Thalmann95]

When the participants are represented by virtual human figures, the articulated structure of the human body introduces a new complexity in the usage of the network resources; because the size of a message needed to represent the body posture is much larger than the one needed for simple, non-articulated objects. This might create a significant overhead in communication, especially as the number of participants in the simulation increases. The animation should also be realistic, as it is easy to recognize anomalies in the virtual human body movements. These requirements should be taken into consideration for human modeling and communication in networked virtual environments. The dead-reckoning technique is a way to decrease the amount of messages communicated among the participants, and has been used for simple non-articulated objects in popular systems. The technique is based on extrapolation of other participants if they have small change in their position. In this paper, we present a dead-reckoning approach for articulated virtual human figures, and discuss main issues and experimental results.

The next section discusses the basic dead reckoning technique. Then, we discuss the virtual human model characteristics that we use in our system. Afterwards, we present the dead-reckoning algorithm for virtual humans and the main issues to consider for this task. Then, we discuss experimental results and future work.

*Published in Proc. VRAIS '97, pp. 161-169

2 Dead Reckoning

The dead-reckoning algorithm is a way to decrease the amount of messages communicated among the participants, and is used for simple non-articulated objects in popular systems such as DIS [IEEE93], NPSNET [Macedonia 94].

To describe the dead-reckoning algorithm, similar to [Gossweiler 94], we can give an example of space dogfight game with n players. Each player is represented by, and can control, a different ship. When a player X moves its own ship, it sends a message to all n-1 players, containing the new position. When all players move once, a total of n*(n-1) messages are communicated. To reduce the communication overhead, the player X sends the ship's position and velocity to other participants. The other participants will use the velocity information to extrapolate the next position of the participant X. This extrapolation operation is named dead-reckoning.

In this approach, each participant also stores another copy of its own model, called *ghost* model, to which it applies dead-reckoning algorithm. If the difference between the real position and this additional copy is greater than a predefined maximum, then player X sends the real position and velocity to other participants, so that they can correct their copy of participant X's object. Note that player X sends messages only if there is a big difference between the real position and extrapolated one.

The performance of the dead-reckoning algorithm is dependent on how it correctly predicts the next frames. Therefore, the characteristics of the simulation, and the body model should be taken into account for developing the algorithm.

3 Virtual Human Representation

Real-time representation and animation of virtual human figures has been a challenging and active area in computer graphics [Boulic 95][Badler 93]. Typically, an articulated structure corresponding to the human skeleton is needed for the control of the body posture. Structures representing the body shape have to be attached to the skeleton, and clothes may be wrapped around the body shape. We use the HUMANOID articulated human body model with 74 degrees of freedom without the hands, with additional 30 degrees

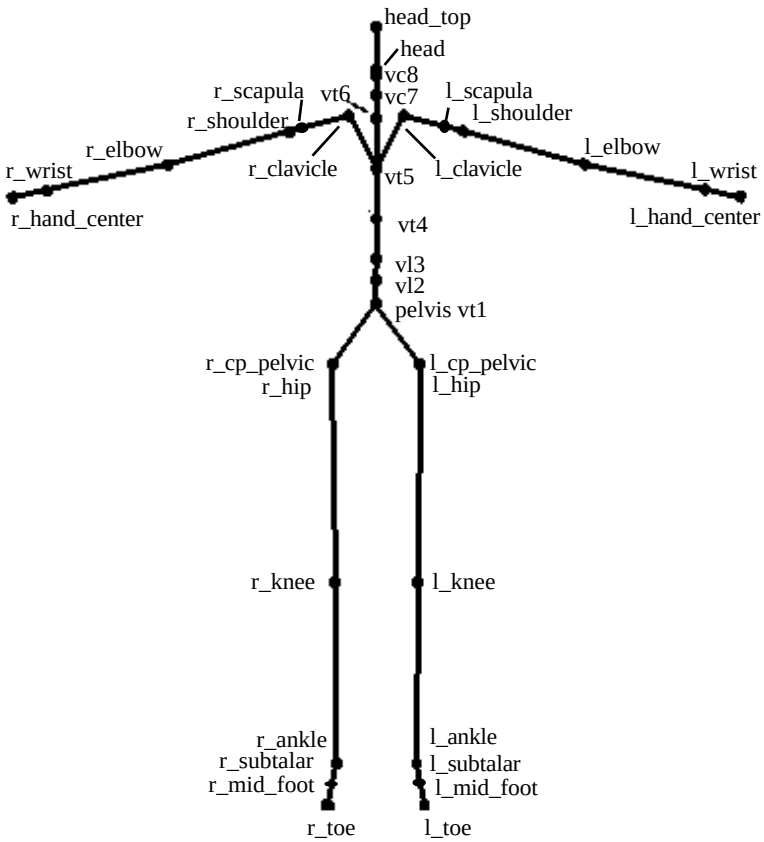

Fig. 1. Virtual Human Figure Representation

of freedom for each hand. In this paper, we focus on the body joints ignoring the hand joints, however the same algorithm can also be applied to the hands. The skeleton is represented by a 3D articulated hierarchy of joints, each with realistic maximum and minimum limits. The skeleton is encapsulated with geometrical, topological, and inertial characteristics of different body limbs. The body structure has a fixed topology template of joints, and different body instances are created by specifying 5 scaling parameters: global scaling, frontal scaling, high and low lateral scaling, and the spine origin ratio between the lower and upper body.

Attached to the skeleton, is a second layer that consists of blobs (metaballs) to represent muscle and skin. The method's main advantage lies in permitting us to cover the entire human body with only a small number of blobs. From this point we divide the body into 17 parts : head, neck, upper torso, lower torso, hip, left and right upper arm, lower arm, hand, upper leg, lower leg, and foot. Because of their complexity, head, hands and feet are not represented with blobs, but instead with triangle meshes. For the other parts a cross-sectional table is used for deformation. This cross-sectional table is created only once for each body by dividing each body part into a number of cross-sections and computing the outermost intersection points with the blobs. These points represent the skin contour and are stored in the body description file. During runtime the skin contour is attached to the skeleton, and at each step is interpolated around the link depending on the joint angles. From this interpolated skin contour the deformation component creates the new body triangle mesh. Thus, the body information in one frame can be represented as the rotational joint angle values.

4 Dead-Reckoning for Virtual Human Figures

The dead-reckoning algorithm on virtual human figures works on their position and body joint angles. There are different possible levels of dead-reckoning of virtual humans:

- *joint-level dead-reckoning:* This approach requires no knowledge on the type of action the figure is executing (e.g. walking, grasping) and no information on the motion control method (e.g. inverse kinematics, dynamics, real-time motion capture). The joint angles of the virtual body are considered to be only available information; and the dead-reckoning computations are performed on this information.
- *action-level dead-reckoning:* The algorithms within this approach know that the actor is performing a particular action (e.g. walking), and parameters of the actor's state (e.g. tired). There have been a few work on the automatic recognition and synthesis of the participants' actions [Unuma95][Tosa96]**.** In this type of dead-reckoning, the algorithm uses the parameters of the current motion (for example, speed and direction for walking); and uses higher level motion control mechanisms (walking motor [Boulic90]) to obtain the motion.

In this paper, we provide a joint-level dead-reckoning algorithm. In order to predict in-between postures between messages, we use a Kalman Filter.

4.1 Kalman Filtering

The Kalman filter is an optimal linear estimator that minimizes the expected mean-square error in the estimated state variables. It provides a means to infer the missing information using noisy measurements, and is used in predicting the future courses of dynamic systems. Its efficient recursive formulation allows the algorithm to keep up with the real-time requirements of posture prediction. For further information on Kalman filtering, see [Brown92]. Previously, in the virtual reality field, the Kalman filter has been applied to decrease the lag between tracking and display in the head trackers, for an overview see [Azuma95][Foxlin96]. In this paper, we present an extension of these methods for the whole body. We assume that the trackers attached to different parts of the body behave independently, therefore the prediction for each joint can be performed independently.

There is considerable freedom in modeling the system, depending on the knowledge of the modeled processes. In our system, we make the following assumptions: at a time frame, only the joint angles of the body are available, and their velocity and acceleration are to be computed. The 74 degrees of freedom can be decomposed into 74 independent 1-dof values, each using a separate predictor. This makes the system simpler. For this, we assume that joint angles change across the prediction interval are small, therefore it is possible to represent rotations by yaw, pitch, roll operations where the order of operations is not important.

Based on these assumptions, we use a discrete linear Kalman filter of Markov-Gaussian type. This allows us to have a simple solution without having any further information or make assumptions on the type of action that the virtual body is performing. Even with the violated assumptions as given above, we observed that the filter performs well. Each joint has associated with it a different predictor. The state vectors contain the value, velocity and acceleration of each degree of freedom in the presence of noise. The velocity and acceleration are estimated using the joint angle values using the Kalman filter.

4.2 Dead Reckoning Algorithm for Virtual Body

The dead-reckoning algorithm is as follows:

```
for each participant p do
        initialize Kalman filters for body p
At each time step do
        for each participant p do
                if (p == mybody) then
                        /* Compute measured body joints -> store in body[p] */
                        body[p] = Measure()
                        /* Compute predicted body joints of local body: */
                        ghost_body = Kalman( ghost_body)
                        /* Compare body[p] and ghost_body joint angles */
                        delta = compare( body[p], ghost_body)
                        if (delta > maximum_allowed_data) then
                                Send message m with body joints of body[p]
                                Copy body[p] joints to ghost_body joints
                        endif
                end
                else
                        if (message m arrived from participant p)
                                Copy message m joint angle contents to body[p]
                        endif
                        body[p] = Kalman( body[p])
                end
        endfor
end
```

Fig. 2. Dead-reckoning algorithm for Virtual Body

Note that, even if the participant Y receives a message from participant X at time frame i, it still performs Kalman filter computations for body X. This is due to the delay between the time participant X body posture is obtained, and the time that participant Y receives the message that contains this information.

One important decision is to select the metrics to compare two body postures. The common practice to compare two postures has been to compare them with eye. However, the dead-reckoning algorithm requires mathematical comparison of joint angles. There are many possibilities to decide on a comparison metric, among them:

1. maximum of differences between corresponding angles:

   max (body[mybody][joint] - ghost_body[joint]) joint = 1 .. 74

2. maximum of differences between corresponding angles, with different a coefficient for each joint

   max (coef(i)*(body[mybody][joint] - ghost_body[joint])) joint = 1..74

3. 3D distance between corresponding joints

Approach 1 assumes that every angle has equal importance for the posture. However, in most cases, some angles have slight effect on the overall posture (for example, in the hand waving posture, the wrist angles have small effect). Approach 2 tries to take this into consideration by assigning a coefficient to each degree of freedom. The third approach uses the 3D position of each joint, and computes the linear distance between corresponding joints. The third approach is expected to achieve better results in providing a metric to compare two postures; because it takes into consideration the 3D positions similar

to comparison by eye, and the overall posture of the body. In the next section, we compare the performance of these actions on example sequences.

5 Experimental Results

We have implemented the presented algorithm, and tried its performance with varying conditions. As representative examples, we selected three actions: a football kick sequence, a jumping sequence, and a hello sequence. Figure 3 shows the three actions, with their joint changes with respect to time. The football kick action consists of a slightly jerky motion, due to the nature of the behavior and the tracking noise. The jumping sequence is more predictive, except the beginning of the action. The hand-wave sequence mainly involves the right arm joints, with various joint behaviors with respect to time.

Figure 4 shows the performance of the dead-reckoning program for the three example actions, with three approaches of posture comparison as discussed in the previous section. The x-axes show the maximum angle difference between corresponding joint angles of local body and ghost body in Figure 4(a) and 4(b), and maximum Euclidean distance between corresponding joints Figure 4(c). The y-axis denotes the percentage of timesteps where the actions caused message transfer, to the whole period of the motion. A percentage of 100% denotes that the dead-reckoning operation has not been performed, and 70 % shows that 30 % of the timesteps were successfully predicted using the dead-reckoning operation.

Figure 5(a) shows the results for the basic algorithm, with varying maximum allowed angle differences between joints. As expected, as the limit increases, the algorithm prediction rate increases, hence the message communication decreases. Figure 5(b) results were taken using approach 2 of posture comparison by decreasing the coefficient of twisting angles, with the assumption that they have less effect on the final posture. Figure 5(c) shows the results of the approach 3 with varying maximum Euclidean distances. The resulting animation was similar to the original motion, when observed with eye; therefore the results show that using distance metric for comparison achieves better performance in dead-reckoning than joint angle comparison. With an error estimate of maximum 15 cm, a 50 % decrease in exchange of messages could be achieved.

6 Conclusions and Future Work

In this paper, we have presented a dead-reckoning algorithm that is based on the Kalman filter, to be used in networked virtual environments with virtual human figures. The obtained results show that, with acceptable errors in the posture information; it is possible to decrease the network communication overhead considerably. It was also shown that the performance of the general-purpose filter is highly based on the characteristics of the instantaneous motion. The work on networking virtual human figures is just beginning; and it is necessary to develop adaptive dead-reckoning algorithms which adjust their parameters depending on the current motion.

Interaction between an a real person and an autonomous virtual actor also requires that the autonomous actor reacts without delay to the real person. This requires prediction of the real actor's movements. The prediction approach presented in this paper can also be used within this application.

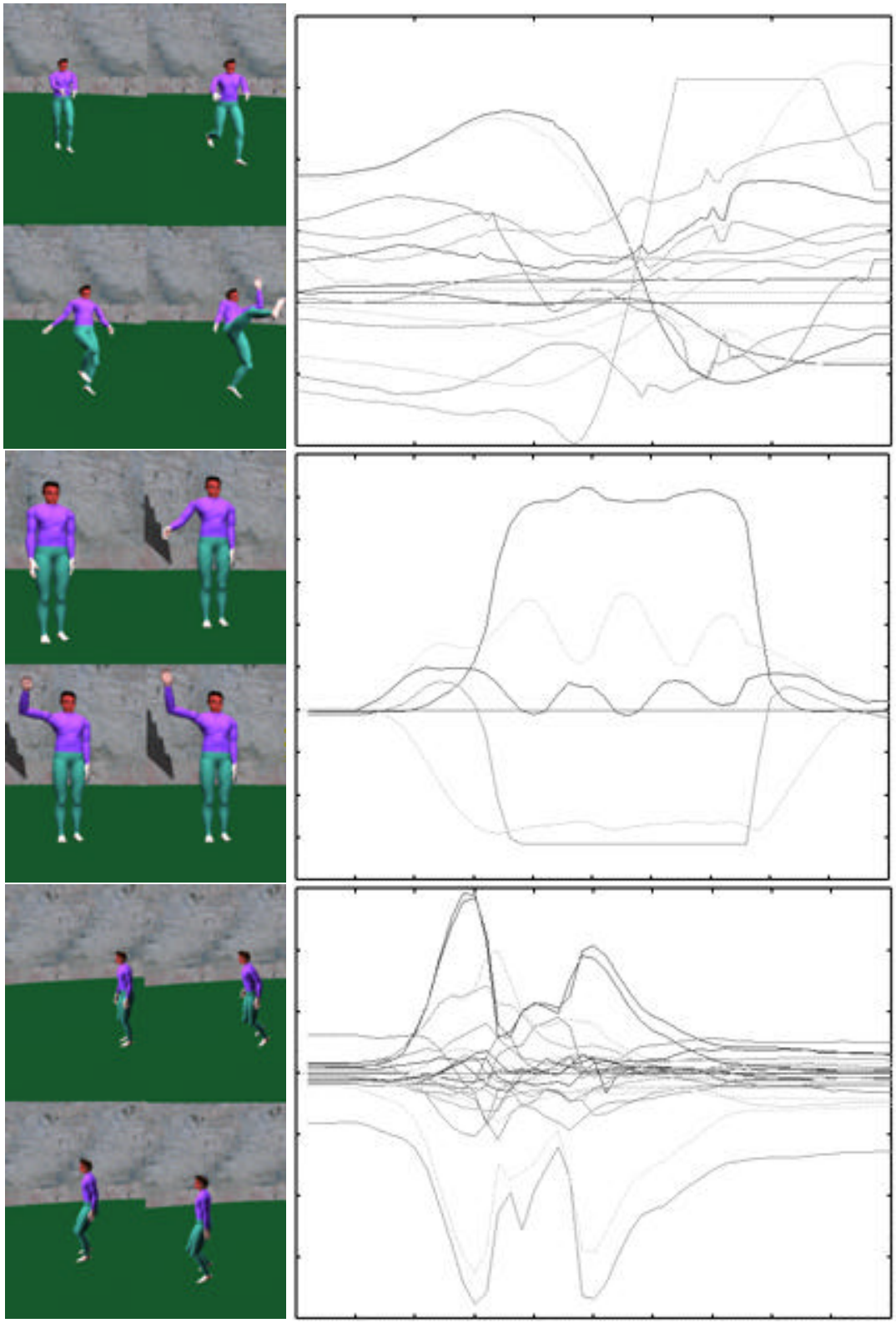

Fig. 3. Example sequences and the joints value changes with respect to time.
(a) football, (b) hello, (c) jump sequence.

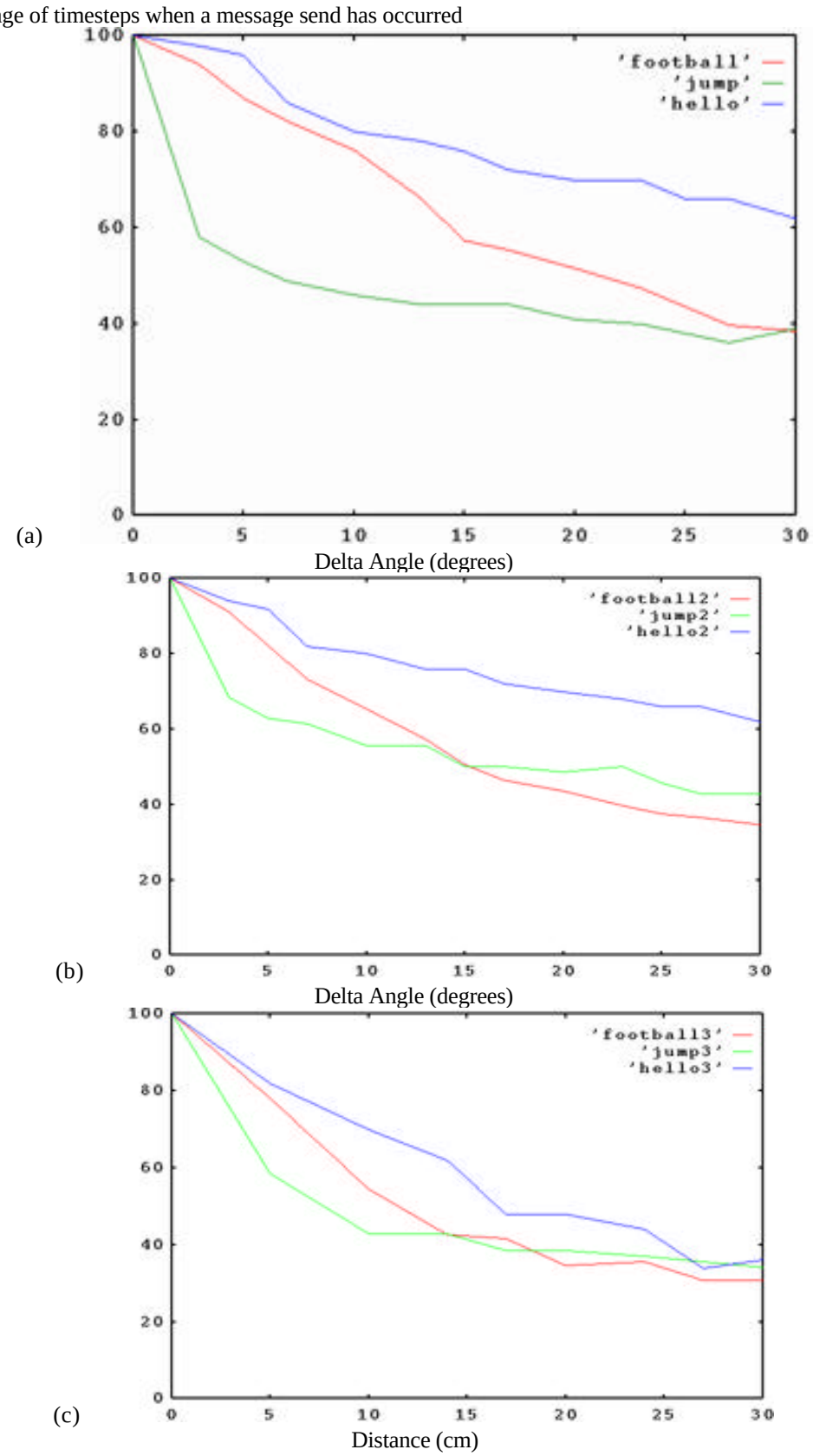

Figure 4 (cont.): Performance of the Kalman Filter with varying delta values. y-axis shows the percentage of message communication.(a) approach 1 for comparison (maximum of joint angle difference), (b) approach 2 (maximum of joint angle difference with corresponding angle coefficients, (c) approach 3 (Euclidean distance between corresponding difference.

Acknowledgments

The authors would like to thank Luc Emering, Tom Molet for their flock of birds interface and motion sequences; Jean-Michel Puiatti for his basic implementation of Kalman filter; and the assistants at LIG and MiraLAB for their contributions in the human models. This work is partially supported by European TEN-IBC VISINET project, and the Swiss SPP Fund.

References

[Azuma95] Ronald Azuma, Gary Bishop, “A Frequency-Domain Analysis of Head-Motion Prediction”, Proc. ACM SIGGRAPH’95, 1995.

[Badler 93] N. I. Badler, C. B. Phillips, B. L. Webber, Simulating Humans: Computer Graphics Animation and Control, Oxford University Press, 1993.

[Boulic 90] Boulic R., Magnenat-Thalmann N. M.,Thalmann D. "A Global Human Walking Model with Real Time Kinematic Personification", *The Visual Computer*, Vol.6(6),1990.

[Boulic 95] R. Boulic et al. "The HUMANOID environment for Interactive Animation of Multiple Deformable Human Characters", Proc. Eurographics'95, 1995.

[Foxlin96] E. Foxlin, “Inertial Head-Tracker Sensor Fusion by a Complementary Separate-Bias Kalman Filter”, Proc. IEEE VRAIS’96.

[Gossweiler 94] Rich Gossweiler, Robert J. Laferriere, Michael L. Keller, Pausch, "An Introductory Tutorial for Developing Multiuser Virtual Environments", Presence: Teleoperators and Virtual Environments, Vol. 3, No. 4, 1994.

[IEEE93] Institute of Electrical and Electronics Engineers, International Standard, ANSI/IEEE Standard 1278-1993, Standard for Information Technology, Protocols for Distributed Interactive Simulation, March 1993.

[Macedonia 94] M.R. Macedonia, M.J. Zyda, D.R. Pratt, P.T. Barham, "NPSNET: A Network Software Architecture for Large-Scale Virtual Environments", Presence: Teleoperators and Virtual Environments, Vol. 3, No. 4, 1994.

[Thalmann95] D. Thalmann, T. K. Capin, N. Magnenat Thalmann, I. S. Pandzic, “Participant, User-Guided, Autonomous Actors in the Virtual Life Network VLNET”, *Proc. ICAT/VRST ’95*, pp. 3-11.

[Tosa 96] Naoko Tosa, Ryohei Nakatsu, “The Esthetics of Artificial Life: Human-Like Communication Character ‘MIC’ and Feeling Improvisation Character ‘MUSE’”, Proc. Artificial Life, 1996.

[Unuma95] Munetoshi Unuma, Ken Anjyo, Ryozo Takeuchi, “Fourier Principles for Emotion-Based Human Figure Animation, Proceedings of ACM SIGGRAPH’95, 1995.